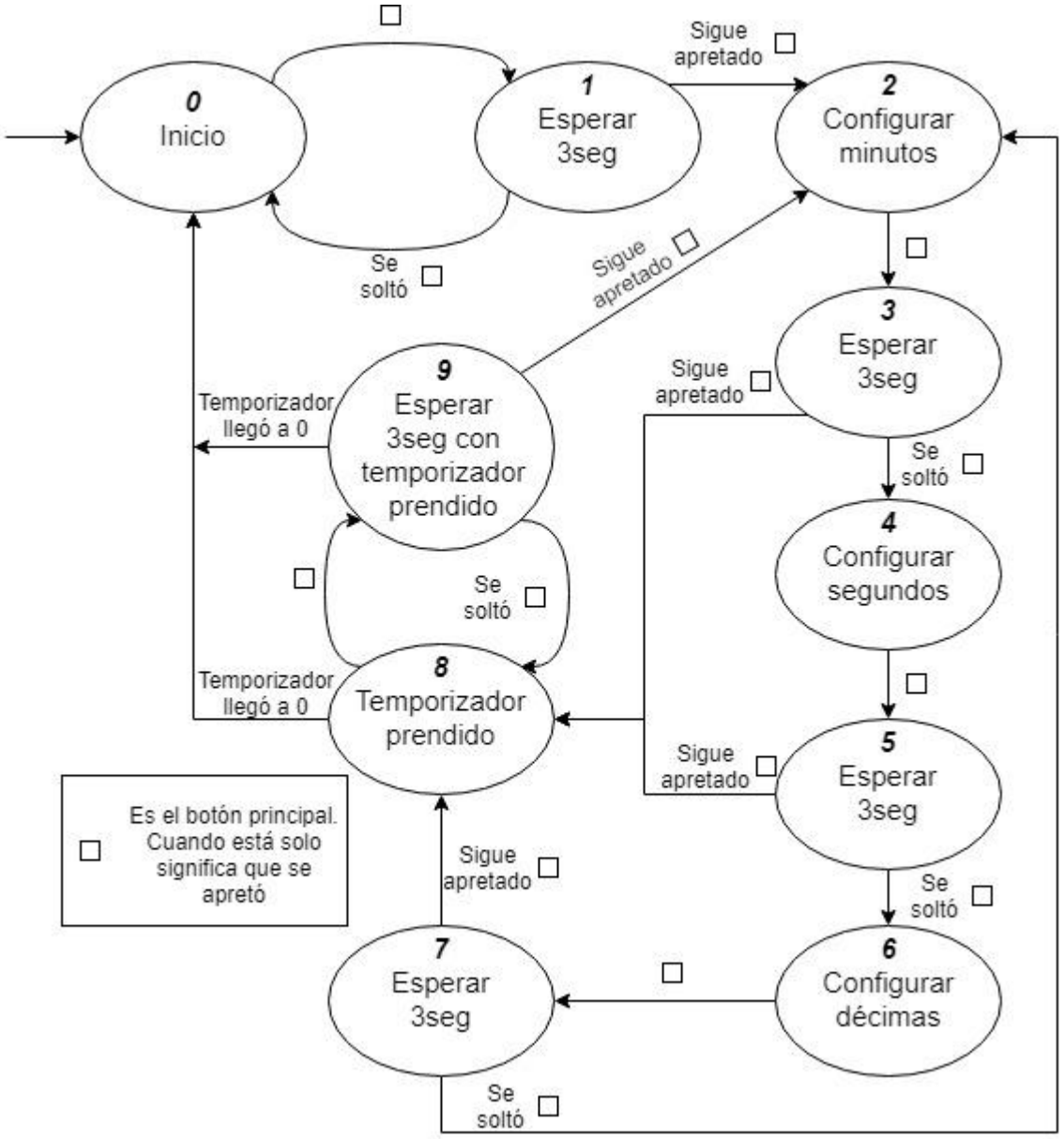
Diseño de temporizador por estados

El principal problema que surge al emplear interrupciones en el programa de un temporizador es que una interrupción puede significar diferentes cosas dependiendo del contexto, y la solución propuesta es usar estados.

¿Qué es un estado?

Es aquello que nos dice qué bloque de código ejecutar. Gracias a ello interpretamos el entorno en el que se encuentra el temporizador con un simple número. Nos dicta qué hacer en caso de interrupciones. Puedo salir del estado actual e ingresar a otro solo si se cumple con una condición que altere la manera que el temporizador tiene para interpretar las interrupciones en determinado momento.



El diagrama anterior muestra todos los estados y las condiciones que pueden resultar en un cambio de estado.

Cada estado nos permite interpretar cada interrupción de una manera particular.

Si no es necesario realizar ninguna acción cuando surge cierta interrupción en un estado determinado, entonces se regresa a donde estaba sin haber cambiado ningún dato. Como si nada hubiera pasado.

Esta manera de afrontar interrupciones aporta múltiples ventajas:

- Evito que una interrupción haga algo que no quiero.
- Es fácil determinar qué código se necesita para cada interrupción dependiendo del estado.
- Es fácil insertar más bloques de código.
- Dado que las funciones que necesita no deben afectar el flujo principal, estas resultan ser muy específicas. Dicha particularidad permite que cada función pueda ser llamada en múltiples ambientes. Todas se podrían implementar en cualquier otro programa cambiando pequeños o nulos aspectos.

Problema del temporizador:

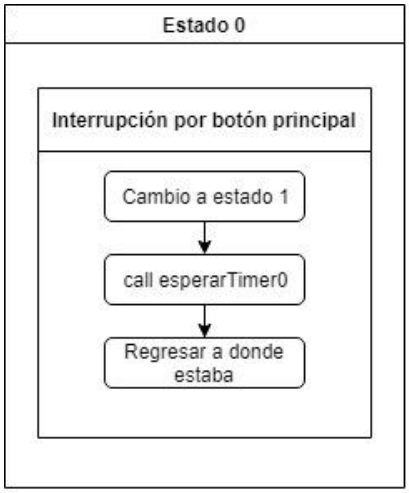
Vamos a programar un temporizador que se controla por 3 botones, se comunica con una pantalla de 8 dígitos siguiendo el protocolo SPI y nos permite visualizar y configurar los minutos, segundos y décimas de segundo que va a tomar. Un botón es el principal, y los otros 2 permiten aumentar o disminuir la unidad de tiempo que queramos modificar. En el estado inicial (0) no puedo configurar nada. Si dejo apretado el botón principal en el estado 0 durante 3 segundos podré configurar las unidades de tiempo. Si dejo apretado el botón por 3 segundos mientras estoy en un estado que me permita configurar alguna unidad de tiempo prendo el temporizador, pero si no lo mantengo apretado a los 3 segundos me cambio de estado para configurar una unidad de tiempo distinta a la actual. También puedo apagar el temporizador si dejo apretado el botón principal durante 3 segundos, y esto me lleva al estado de configurar minutos conservando el valor que tenían las unidades de tiempo al apagar el temporizador.

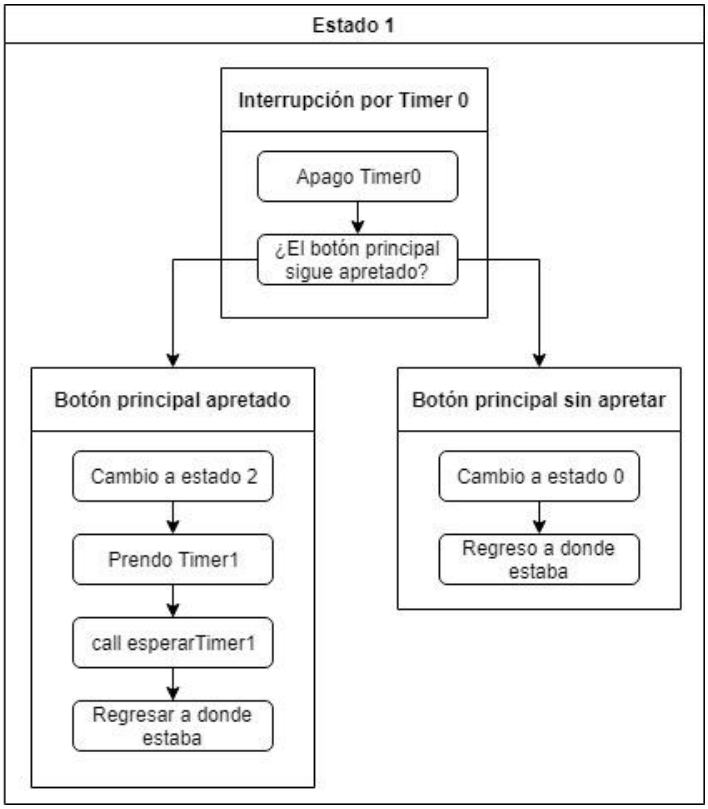
Solución usando estados:

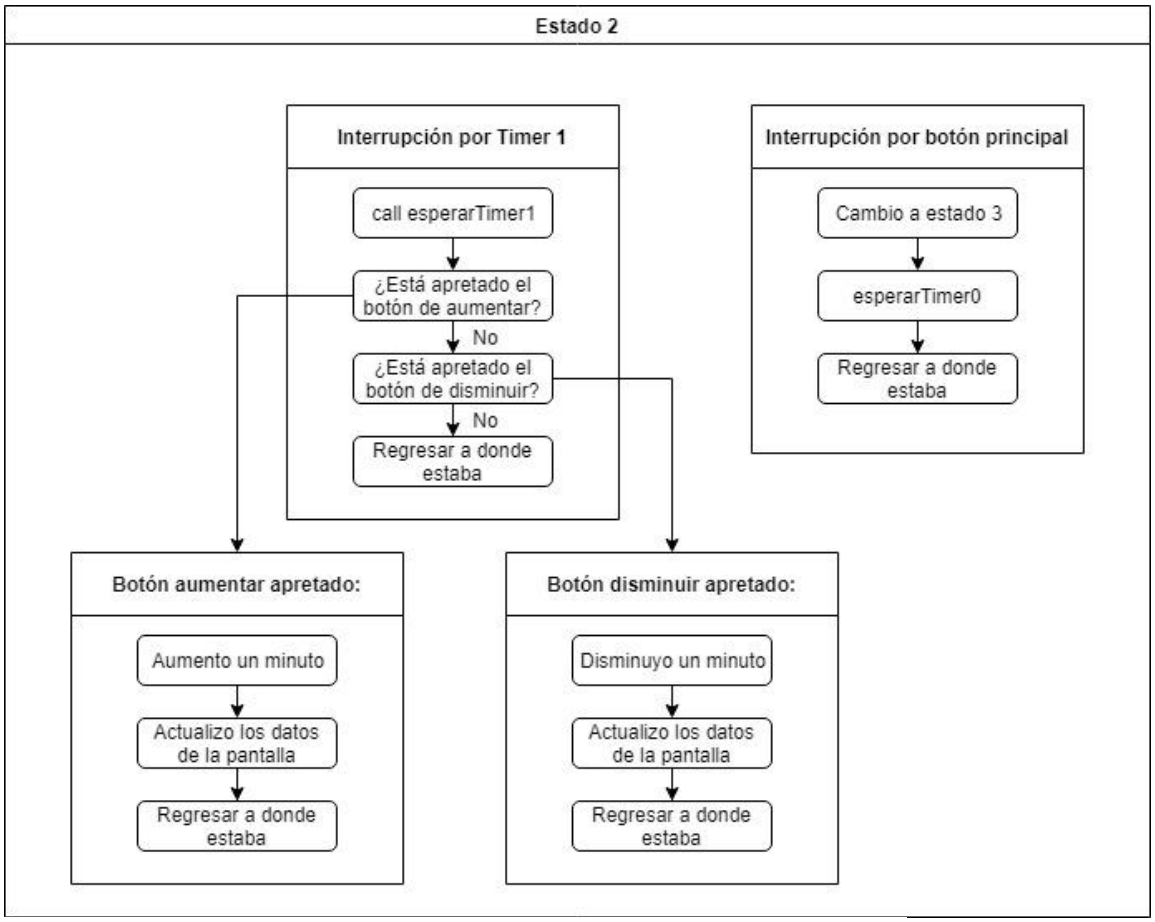
El microchip que se usará es el PIC18F4550. Vamos a destinar el Timer0 para medir 3 segundos y el Timer1 para medir una décima de segundo. No podemos hacer que cada botón genere una interrupción porque el microchip solo tiene 2 patitas que activan interrupciones externas y nosotros usamos 3 botones. La solución es que el botón principal sea el único que genere interrupciones. Si estamos en un estado que nos permita configurar las unidades de tiempo, vamos a preguntar cada décima de segundo si se está apretando algún botón (preguntando

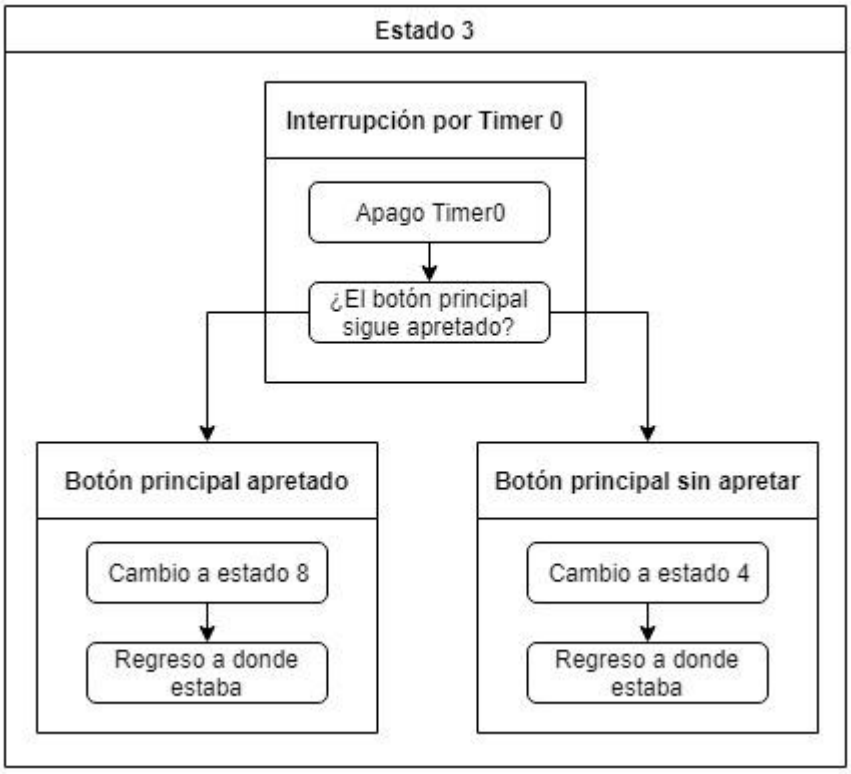
primero si está apretado el de aumentar y luego si está el de disminuir). Esta solución también permite aumentar o disminuir valores grandes si dejo apretado el botón. La desventaja es que es muy rápido, ya que cada segundo que lo deje apretado la variable se va a aumentar en 10. Esta pequeña desventaja tiene soluciones fáciles pero dejaremos el código así porque nos urge completarlo. El botón principal está conectado al pin 2 del puerto B, y los botones de aumentar y disminuir están conectados a los pines 0 y 1, respectivamente, del puerto D. Los botones van a mandar "1" a las respectivas patitas cuando se aprieten.

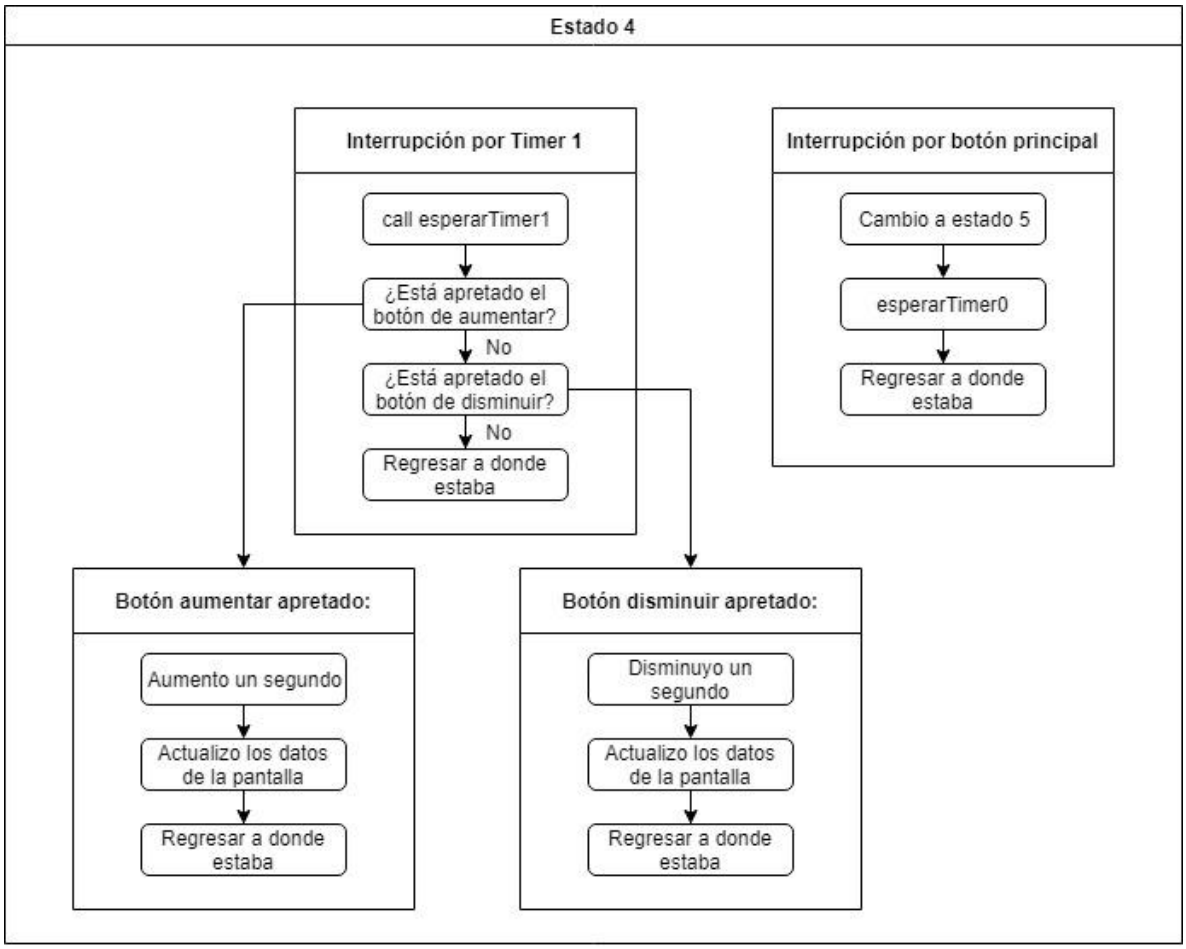
Acciones que realizan las interrupciones dependiendo del estado:

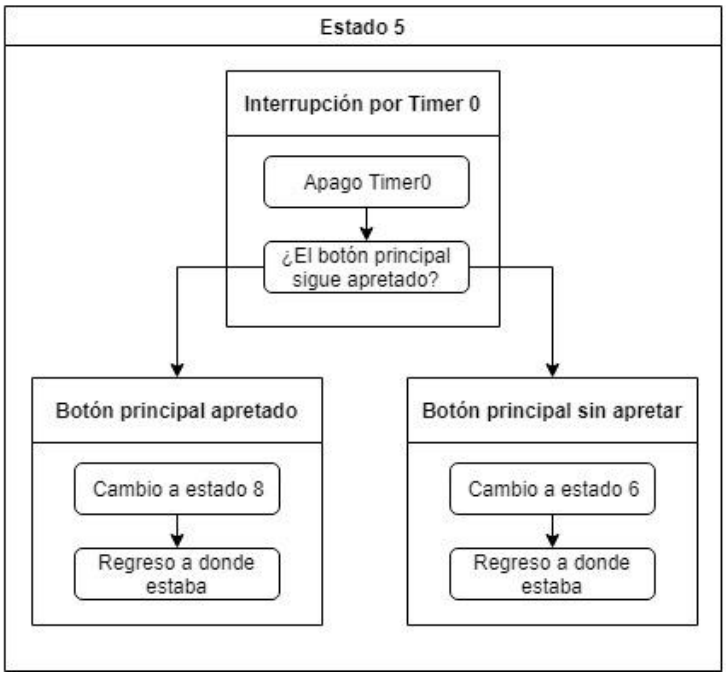


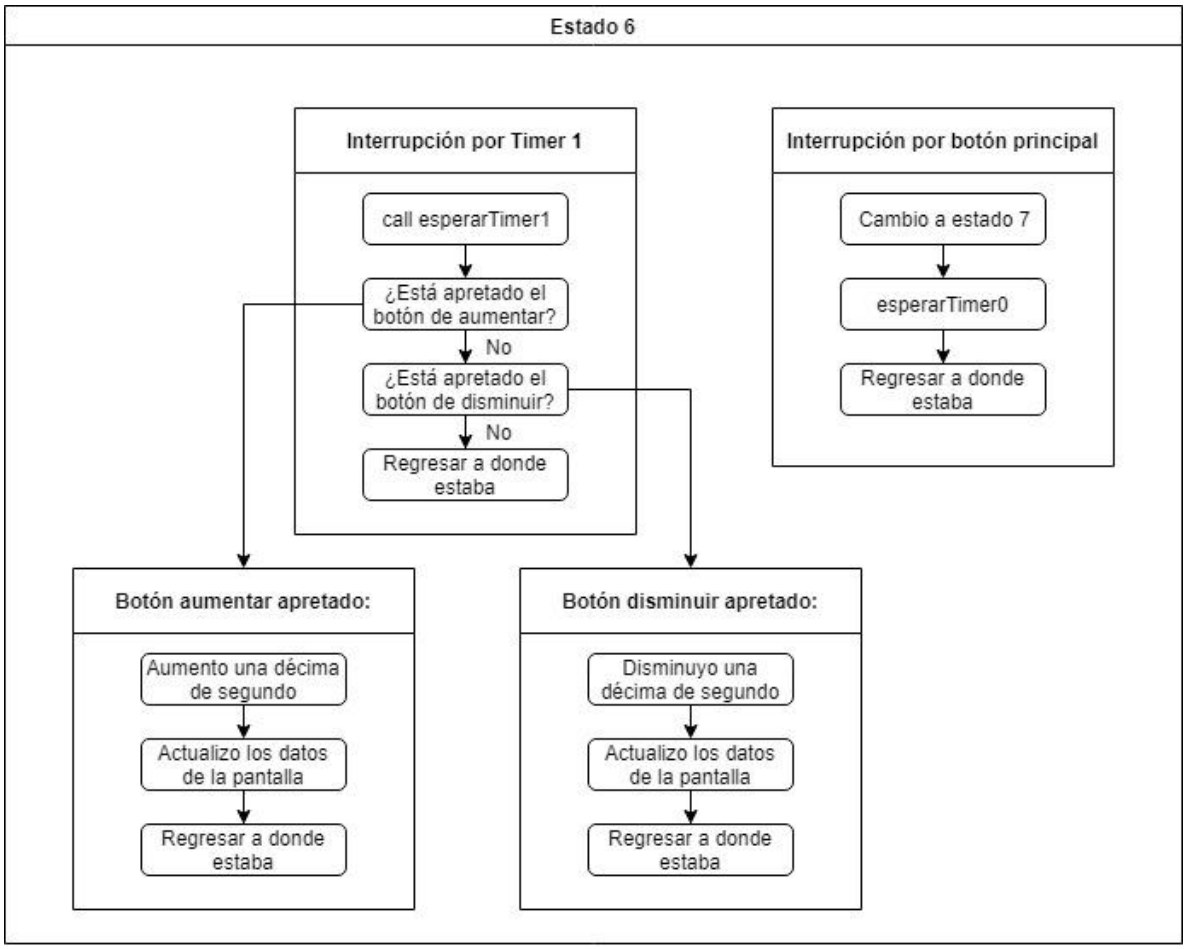


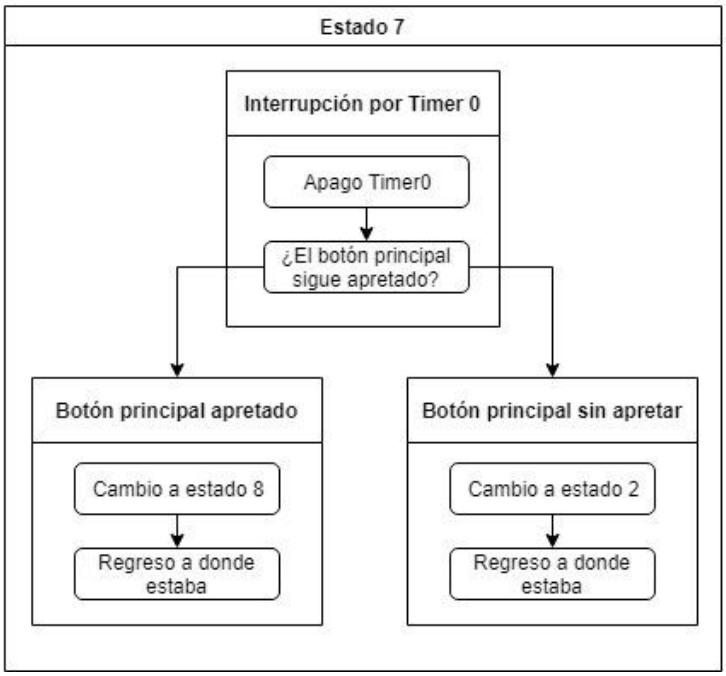

Estado 2
Interrupción por Timer 1
call esperarTimer1
¿Está apretado el botón de aumentar?
No
¿Está apretado el botón de disminuir?
No
Regresar a donde estaba
Interrupción por botón principal
Cambio a estado 3
esperarTimer0
Regresar a donde estaba
Botón aumentar apretado:
Aumento un minuto
Actualizo los datos de la pantalla
Regresar a donde estaba
Botón disminuir apretado:
Disminuyo un minuto
Actualizo los datos de la pantalla
Regresar a donde estaba

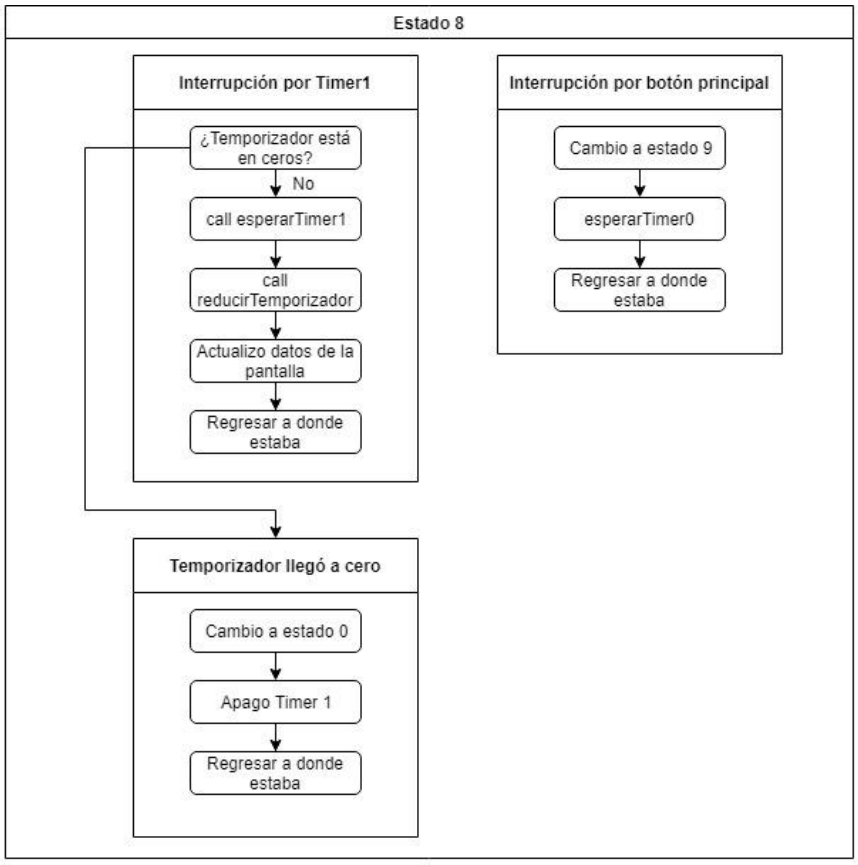

Estado 3
Interrupción por Timer 0
Apago Timer0
¿El botón principal sigue apretado?
Botón principal apretado
Cambio a estado 8
Regreso a donde estaba
Botón principal sin apretar
Cambio a estado 4
Regreso a donde estaba

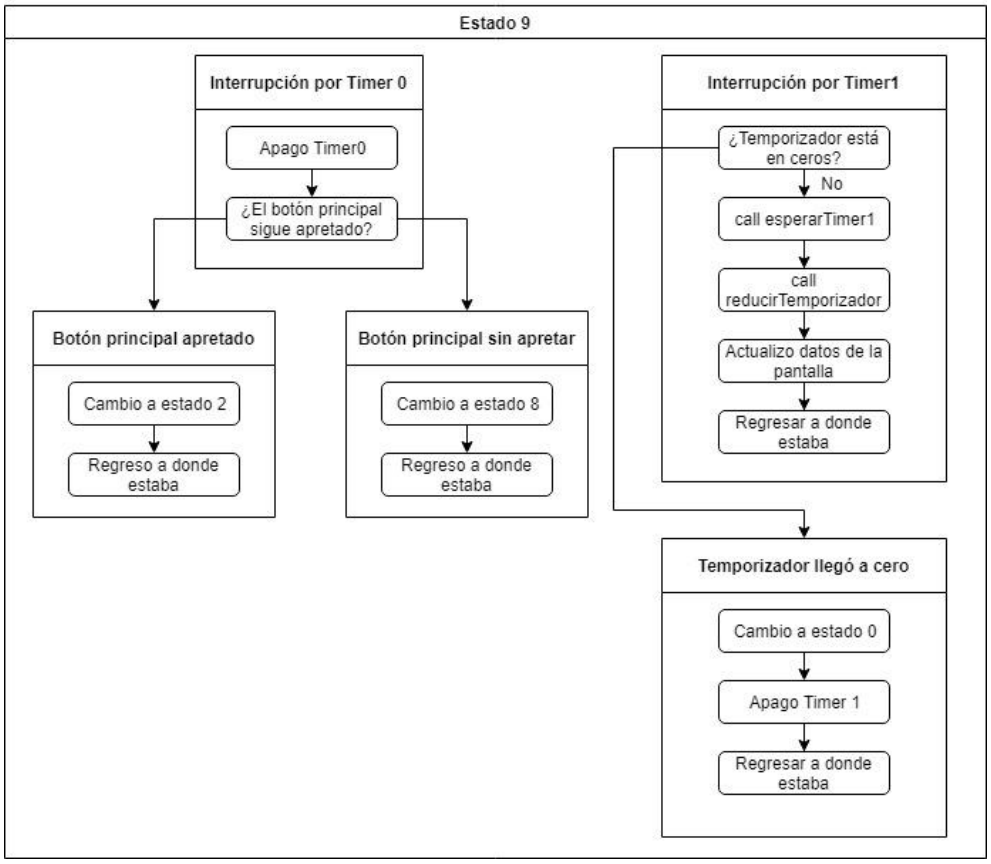

Estado 4
Interrupción por Timer 1
call esperarTimer1
¿Está apretado el botón de aumentar?
No
¿Está apretado el botón de disminuir?
No
Regresar a donde estaba
Interrupción por botón principal
Cambio a estado 5
esperarTimer0
Regresar a donde estaba
Botón aumentar apretado:
Aumento un segundo
Actualizo los datos de la pantalla
Regresar a donde estaba
Botón disminuir apretado:
Disminuyo un segundo
Actualizo los datos de la pantalla
Regresar a donde estaba


Estado 5
Interrupción por Timer 0
Apago Timer0
¿El botón principal sigue apretado?
Botón principal apretado
Cambio a estado 8
Regreso a donde estaba
Botón principal sin apretar
Cambio a estado 6
Regreso a donde estaba


Estado 6
Interrupción por Timer 1
call esperarTimer1
¿Está apretado el botón de aumentar?
No
¿Está apretado el botón de disminuir?
No
Regresar a donde estaba
Interrupción por botón principal
Cambio a estado 7
esperarTimer0
Regresar a donde estaba
Botón aumentar apretado:
Aumento una décima de segundo
Actualizo los datos de la pantalla
Regresar a donde estaba
Botón disminuir apretado:
Disminuyo una décima de segundo
Actualizo los datos de la pantalla
Regresar a donde estaba


Estado 7
Interrupción por Timer 0
Apago Timer0
¿El botón principal sigue apretado?
Botón principal apretado
Cambio a estado 8
Regreso a donde estaba
Botón principal sin apretar
Cambio a estado 2
Regreso a donde estaba


Estado 8
Interrupción por Timer1
¿Temporizador está en ceros?
No
call esperarTimer1
call reducirTemporizador
Actualizo datos de la pantalla
Regresar a donde estaba
Interrupción por botón principal
Cambio a estado 9
esperarTimer0
Regresar a donde estaba
Temporizador llegó a cero
Cambio a estado 0
Apago Timer 1
Regresar a donde estaba


Estado 9
Interrupción por Timer 0
Apago Timer0
¿El botón principal sigue apretado?
Botón principal apretado
Cambio a estado 2
Regreso a donde estaba
Botón principal sin apretar
Cambio a estado 8
Regreso a donde estaba
Interrupción por Timer1
¿Temporizador está en ceros?
No
call esperarTimer1
call reducirTemporizador
Actualizo datos de la pantalla
Regresar a donde estaba
Temporizador llegó a cero
Cambio a estado 0
Apago Timer 1
Regresar a donde estaba

El problema de esta solución es que cuando se está configurando alguna unidad de tiempo y se aprieta el botón principal para cambiar de campo debo esperar 3 largos segundos para poder seguir cambiando los datos.

Lista de funciones:

- configEntrada: **Terminada**
  - Configuro los pines que van a usar los botones como entrada. Configuro que desde el pin al que está conectado el botón principal se puedan generar interrupciones de prioridad baja con flanco de subida.
- configurarDriver: **Terminada**
  - Configuro todo lo necesario para que el driver funcione
- configTimer0: **Terminada**
  - Habilito la interrupción del Timer0, le asigno prioridad la prioridad baja y bajo la bandera. No lo prendo.
- configTimer1: **Terminada**
  - Habilito la interrupción del Timer1, le asigno prioridad la prioridad baja y bajo la bandera. No lo prendo.
- activarInterrupciones: **Terminada**
  - Activo interrupciones.
- prenderPantalla: **Terminada**
  - Prendo la pantalla.
- actualizarDatos: **Terminada**
  - Actualizo los datos de la pantalla
- prenderTimer0: **Terminada**
  - Prendo el Timer0.
- apagarTimer0: **Terminada**
  - Apago el Timer0.
- prenderTimer1: **Terminada**
  - Prendo el Timer1.
- apagarTimer1: **Terminada**
  - Apago el Timer1.
- esperarTimer0: **Terminada**
  - Bajo la bandera del timer 0, le asigno el valor apropiado para que pasen 3 segundos y prendo el timer 0.
- esperarTimer1: **Terminada**
  - Bajo la bandera del timer 1 y le asigno el valor apropiado para que pase una décima de segundo.
- aumento_minuto: **Terminada**
  - Aumento el valor de la variable "minutos" en uno, verificando que no se pase de 59.
- disminuyo_minuto: **Terminada**
  - Disminuyo el valor de la variable "minutos" en uno, verificando que no sea menos que 0.

- aumento_segundos: **Terminada**
  - Aumento el valor de la variable "segundos" en uno, verificando que no se pase de 59.
- disminuyo_segundos: **Terminada**
  - Disminuyo el valor de la variable "segundos" en uno, verificando que no sea menos que 0.
- aumento_deciseg: **Terminada**
  - Aumento el valor de la variable "decimas" en uno, verificando que no se pase de 9.
- disminuyo_deciseg: **Terminada**
  - Disminuyo el valor de la variable "decimas" en uno, verificando que no sea menor que 0.
- reducirTemporizador: **Terminada**
  - Disminuyo la variable "decimas" en uno cada vez que se llame. Si "decimas" es igual a cero, disminuye la variable "segundos" en uno. Si "segundos" es igual a cero, disminuye la variable "minutos" en uno. No es necesario que verifique cuando minutos llegue a cero.
- checarSi0: **Terminada**
  - Checo si tanto "minutos", "segundos" y "decimas" valen cero. En caso de que valgan cero, asignar el valor de 1 a la variable "tempVale0".
- separacion_minuto: **Terminada**
  - Guardo las decenas de "minutos" en la variable "minutodec", y las unidades en "minutounidades". No modifico el valor de "minutos".
  - Esta función no es llamada desde el flujo principal, pero le sirve a la función "actualizarDatos".
- separacion_segundo: **Terminada**
  - Guardo las decenas de "segundos" en la variable "segundodec", y las unidades en "segundounidades". No modifico el valor de "segundos".
  - Esta función no es llamada desde el flujo principal, pero le sirve a la función "actualizarDatos".
- iniciarMinMax: **Terminada**
  - Inicia las variables que definen los valores máximos y mínimos que van a tener los minutos, segundos y las décimas de segundos.

Estas funciones deben tener disponibles las siguientes variables universales:

- minutos
- segundos
- decimas
- minutodec
- minutounidades
- segundodec
- segundounidades
- tempVale0
- digito
- altoBajo